Finally, the workload monitor shall update the following parameters for use by admission controller 424, for example, using an I/O admission control algorithm that includes a resource model such as Resource Model Equation (19):

$$MaxNoV\_perDisk = \max \{ \text{TotalNov(i), for all disk drive i} \} \quad (29)$$

$$MaxAggRate\_perDisk = \max \{ \text{TotalRate(i), for all disk drive i} \} \quad (30A)$$

Although equations of the previously described embodiment may be implemented in a storage environment employing sub-disk partitioning, it will be understood that in other embodiments the sub-disk layer may be disabled if so desired so that a partition of storage devices (*e.g.*, physical disk drive) into sub-disks is not allowed.

Multiple Storage Device Buffer Allocation in Substantially-Unbalanced Workload Environments

To further address unbalanced workload distribution, an optional multiple device buffer allocation scheme may be employed in one embodiment to help optimize buffer space utilization. FIG. 4B shows maximal available buffer memory space ("B_{max}") 200, and multiple storage devices 210, 212, 214, 216 and 218 that may be, for example, individual disk drives. In this example, B_{max} is 1.0 GB. Workload weight per storage device is also shown in FIG. 4B and is expressed as a percentage of total workload, *i.e.*, 50% for device 210, 30% for device 212, 10% for each of respective devices 214 and 216, and 0% for device 218. In this regard workload weight for each storage device may be calculated or estimated using any suitable method such as, for example, by using "CurrentWeight" formula (24) previously given:

$$\text{CurrentWeight(i)} = \alpha * \text{CurrentWeight(i)} + (1 - \alpha) * \text{NewWeight(i)} \quad (24)$$

Once workload weight for each storage device has been calculated or estimated, buffer memory space 200 may be logically or soft-partitioned into individual buffer memory spaces

B(i) assigned to each storage device or group of storage devices (*e.g.,* logical volume group, tenant group, CoS group, *etc.*) (i) (*e.g.,* 210, 212, 214, 216 and 218). In this regard, a portion of maximal available buffer memory space B_{max} may be allocated to a respective storage device (i) in a manner that is at least partially dependent on the value of workload weight determined for that storage device, either proportional to value of workload weight for that device, or non-proportional but dependent on the value of workload weight for that storage device. For proportional allocation, the following formula may be employed to calculate :

$$B(i) = B_{max} * \text{CurrentWeight}(i) \qquad (30B)$$

Thus, as shown in FIG. 4B, storage device 210 is allocated 50% of B_{max} (0.5 GB), storage device 212 is allocated 30% of B_{max} (0.3 GB), storage device 214 is allocated 10% of B_{max} (0.1 GB), storage device 216 is allocated 10% of B_{max} (0.1 GB), and storage device 218 is not allocated any of B_{max}.

Following allocation of buffer memory space, total number of viewers per storage device ("TotalNov_disk(i)"), and total rate per storage device ("TotalRate_disk(i)") may be calculated using equations (27) and (28) previously given:

$$\text{TotalNov\_disk}(i) = \text{Sum of TotalNov\_subdisk}(j) \quad \text{for all subdisk } j \text{ in disk } i \qquad (27)$$

$$\text{TotalRate\_disk}(i) = \text{Sum of TotalRate\_subdisk}(j) \text{ for all subdisk } j \text{ in disk } i \qquad (28)$$

Once values of TotalNov_disk(**i**) and TotalRate_disk(**i**) have been calculated above, values of cycle time T(i) for each storage device (i) may be calculated in a manner described for other Resource Model Equations given herein, for example, using the upper bound(i) and lower bound(i) based on the following Resource Model Equation (30C), that is similar to Resource Model Equation (13):

$$TotalNov\_disk(i) * AA \,/\, [1 - Reserved\_Factor - (TotalRate\_disk(i))/\, TR] \leq T(i)$$

$$\leq (1 - Reserved\_Factor) * B_i \,/\, \{Buffer\_Multiplcity * TotalRate\_disk(i)]\} \qquad (30C)$$

Read-ahead size(i) for each storage device (i) may then be calculated, for example, using the calculated value of T(i), block size BL, and play rate or data consumption rate $\mathbf{P}_i$, using the following relationship:

$$\text{Read-ahead size(i)} = [T(i) * \mathbf{P}_i] / BL \tag{30D}$$

Validation of System I/O performance characteristics

Although information management system I/O performance characteristic values may be assumed for each storage device and/or assumed constant for all storage devices (*e.g.*, all disk drives in a multiple disk drive implementation) installed or coupled to a storage processing engine, one embodiment of the disclosed methods and systems may be implemented to perform optional validation of assumed system I/O performance characteristics. Examples of assumed system I/O performance characteristics that may be optionally validated include, but are not limited to, estimated values of seek and rotation latency (*e.g.*, average access time AA), and/or estimated transfer rate (*e.g.*, TR). Optional validation of assumed system I/O performance characteristics may be advantageously employed to optimize information management system I/O performance when assumed performance characteristics are inaccurate. Such may be the case, for example, when assumed values of system I/O performance characteristics are entered wrong, when one or more wrong disk drives are coupled to a system by operational personnel., when an upgrade for disk drives is performed but the configuration table was not updated, when assumed performance characteristics supplied by disk manufacturer are incorrect, *etc*.

In one embodiment, validation of assumed system I/O performance characteristics such as average access time and transfer rate may be implemented before a storage device (*e.g.*, disk drive) is put into service, for example, by reserving a portion of the processing window for running a utility in the storage management processing engine that performs the validation every time the information management system is booted. Such a storage device performance validation may be implemented using an algorithm or routine. For example, in an information